

# HW3 Hadoop Mapreduce

r08942088 李政旻

1. (a) (90%) Write a map reduce application using Python to do the log analysis of apache2 web server access log

- 1.1. 首先使用 google cloud platform 的 dataproc 來建立 cluster, 啟用 dataproc api 後, 選擇標準的一個主要和預設兩個工作站的組合, 由於配額的限制, 所以兩個工作站都只使用一個 CPU, 而主要節點有4個CPU, 並選擇ubuntu做作業系統。

- 1.2. 建立好 cluster 之後, 查看三個機器分別的外部IP, 再使用 SSH進入機器確認環境是否正常。

Google Cloud Platform Cloud Computing 2020 搜尋產品和資源

Compute Engine VM 執行個體 建立執行個體 匯入 VM 重新整理 開始/繼續 停止 暫停 重設

VM 執行個體

篩選 VM 執行個體 資料欄

名稱	區域	建議	使用者	內部 IP	外部 IP	連接
<input type="checkbox"/> cluster-d98c-m	us-central1-c			10.128.0.7 (nic0)	35.184.118.145	SSH
<input type="checkbox"/> cluster-d98c-w-0	us-central1-c			10.128.0.8 (nic0)	35.222.58.49	SSH
<input type="checkbox"/> cluster-d98c-w-1	us-central1-c			10.128.0.6 (nic0)	173.255.116.77	SSH
<input type="checkbox"/> docker-1	us-central1-a			10.128.0.2 (nic0)	35.232.79.119	SSH

```
A520M-4750G@cluster-d98c-m:~$ hadoop envvars
JAVA_HOME='/usr/lib/jvm/adoptopenjdk-8-hotspot-amd64'
HADOOP_COMMON_HOME='/usr/lib/hadoop'
HADOOP_COMMON_DIR='./'
HADOOP_COMMON_LIB_JARS_DIR='lib'
HADOOP_COMMON_LIB_NATIVE_DIR='lib/native'
HADOOP_CONF_DIR='/etc/hadoop/conf'
HADOOP_TOOLS_PATH='/usr/lib/hadoop/share/hadoop/tools/lib/'
```

1.3. 使用SFTP將需要的檔案上傳 (access.log, mapper.py, reducer.py, test.sh)

1.3.1. access.log: 即

[http://hpc.ee.ntu.edu.tw/html/IntelligentClouds/webAccessLog/access\\_log](http://hpc.ee.ntu.edu.tw/html/IntelligentClouds/webAccessLog/access_log)

1.3.2. mapper.py:

```
1  #!/usr/bin/env python
2  """mapper.py"""
3
4  import sys
5  import datetime
6
7  # input comes from STDIN (standard input)
8  for line in sys.stdin:
9      # remove leading and trailing whitespace
10     line = line.strip()
11
12     # extract record time
13     words = line.split("[")
14     if len(words) > 1:
15         record_time = words[1].split()
16     else:
17         continue
18
19     # change time format
20     formatted_record_time = datetime.datetime.strptime(record_time[0], "%d/%b/%Y:%H:%M:%S")
21     print (formatted_record_time.strftime('%Y-%m-%d T %H:00:00.000')+ "\t1")
22
```

### 1.3.3. reducer.py:

```
1  #!/usr/bin/env python
2  """reducer.py"""
3
4  from operator import itemgetter
5  import sys
6
7  current_word = None
8  current_count = 0
9  word = None
10
11  # input comes from STDIN
12  for line in sys.stdin:
13      # remove leading and trailing whitespace
14      line = line.strip()
15
16      # parse the input we got from mapper.py
17      word, count = line.split('\t', 1)
18
19      # convert count (currently a string) to int
20      try:
21          count = int(count)
22      except ValueError:
23          # count was not a number, so silently
24          # ignore/discard this line
25          continue
26
27      # this IF-switch only works because Hadoop sorts map output
28      # by key (here: word) before it is passed to the reducer
29      if current_word == word:
30          current_count += count
31      else:
32          if current_word:
33              # write result to STDOUT
34              print('%s\t%s' % (current_word, current_count))
35          current_count = count
36          current_word = word
37
38  # do not forget to output the last word if needed!
39  if current_word == word:
40      print('%s\t%s' % (current_word, current_count))
41
```

### 1.3.4. test.sh:

```
1 #!/bin/bash
2 # Hadoop stream jar
3 STREAMJAR=/usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.0.jar
4 # input file
5 INPUT=access.log
6 # input directory
7 INPUT_DIR=/input
8 # output file
9 OUTPUT=result.dat
10 # output directory
11 OUTPUT_DIR=/output
12 # mapper file
13 MAPPER=./mapper.py
14 # reducer file
15 REDUCER=./reducer.py
16 # create input directory on hdfs
17 hdfs dfs -mkdir /input
18 # upload input file to input directory
19 hdfs dfs -put $INPUT $INPUT_DIR
20 # remove old output directory
21 hdfs dfs -rm -r -f $OUTPUT_DIR
22 # execute map-reduce with Hadoop stream jar
23 hadoop jar $STREAMJAR -files $MAPPER,$REDUCER -mapper "python $MAPPER" -reducer "python $REDUCER" -input $INPUT_DIR -output $OUTPUT_DIR
24 # download the output file from hdfs
25 hdfs dfs -cat $OUTPUT_DIR/part* | sort -kl,1 > $OUTPUT
26
```

### 1.4. 執行 "bash test.sh"

```
A520M-4750G@cluster-d98c-m:~/hw3$ bash test.sh
packageJobJar: [ [/usr/lib/hadoop-mapreduce/hadoop-streaming-2.10.0.jar] /tmp/streamjob2293235875939410057.jar tmpDir=null
20/11/03 18:17:29 INFO client.RMProxy: Connecting to ResourceManager at cluster-d98c-m/10.128.0.7:8032
20/11/03 18:17:29 INFO client.AHSProxy: Connecting to Application History server at cluster-d98c-m/10.128.0.7:10200
20/11/03 18:17:30 INFO client.RMProxy: Connecting to ResourceManager at cluster-d98c-m/10.128.0.7:8032
20/11/03 18:17:30 INFO client.AHSProxy: Connecting to Application History server at cluster-d98c-m/10.128.0.7:10200
20/11/03 18:17:30 INFO mapred.FileInputFormat: Total input files to process : 1
20/11/03 18:17:30 INFO mapreduce.JobSubmitter: number of splits:15
20/11/03 18:17:31 INFO Configuration.deprecation: yarn.resourcemanager.system-metrics-publisher.enabled is deprecated. Instead, use yarn.system-metrics-publisher.enabled
20/11/03 18:17:31 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1604426318533_0001
20/11/03 18:17:31 INFO conf.Configuration: resource-types.xml not found
20/11/03 18:17:31 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
20/11/03 18:17:31 INFO resource.ResourceUtils: Adding resource type - name = memory-mb, units = Mi, type = COUNTABLE
20/11/03 18:17:31 INFO resource.ResourceUtils: Adding resource type - name = vcores, units = , type = COUNTABLE
20/11/03 18:17:32 INFO impl.YarnClientImpl: Submitted application application_1604426318533_0001
20/11/03 18:17:32 INFO mapreduce.Job: The url to track the job: http://cluster-d98c-m:8088/Proxy/application_1604426318533_0001/
20/11/03 18:17:32 INFO mapreduce.Job: Running job: job_1604426318533_0001
20/11/03 18:17:48 INFO mapreduce.Job: Job job_1604426318533_0001 running in uber mode : false
20/11/03 18:17:48 INFO mapreduce.Job: map 0% reduce 0%
20/11/03 18:18:01 INFO mapreduce.Job: map 13% reduce 0%
20/11/03 18:18:11 INFO mapreduce.Job: map 33% reduce 0%
20/11/03 18:18:13 INFO mapreduce.Job: map 47% reduce 0%
20/11/03 18:18:25 INFO mapreduce.Job: map 60% reduce 0%
20/11/03 18:18:27 INFO mapreduce.Job: map 80% reduce 0%
20/11/03 18:18:33 INFO mapreduce.Job: map 87% reduce 0%
20/11/03 18:18:36 INFO mapreduce.Job: map 100% reduce 0%
20/11/03 18:18:44 INFO mapreduce.Job: map 100% reduce 100%
20/11/03 18:18:47 INFO mapreduce.Job: Job job_1604426318533_0001 completed successfully
20/11/03 18:18:48 INFO mapreduce.Job: Counters: 49
  File System Counters
    FILE: Number of bytes read=46392
    FILE: Number of bytes written=3824095
```



```
20/11/03 18:18:27 INFO mapreduce.Job: map 80% reduce 0%
20/11/03 18:18:33 INFO mapreduce.Job: map 87% reduce 0%
20/11/03 18:18:36 INFO mapreduce.Job: map 100% reduce 0%
20/11/03 18:18:44 INFO mapreduce.Job: map 100% reduce 100%
20/11/03 18:18:47 INFO mapreduce.Job: Job job_1604426318533_0001 completed successfully
20/11/03 18:18:48 INFO mapreduce.Job: Counters: 49
```

#### File System Counters

```
FILE: Number of bytes read=46392
FILE: Number of bytes written=3824095
FILE: Number of read operations=0
FILE: Number of large read operations=0
FILE: Number of write operations=0
HDFS: Number of bytes read=233143
HDFS: Number of bytes written=2825
HDFS: Number of read operations=55
HDFS: Number of large read operations=0
HDFS: Number of write operations=6
```

#### Job Counters

```
Launched map tasks=15
Launched reduce tasks=2
Data-local map tasks=15
Total time spent by all maps in occupied slots (ms)=784232
Total time spent by all reduces in occupied slots (ms)=87640
Total time spent by all map tasks (ms)=196058
Total time spent by all reduce tasks (ms)=10955
Total vcore-milliseconds taken by all map tasks=196058
Total vcore-milliseconds taken by all reduce tasks=10955
Total megabyte-milliseconds taken by all map tasks=200763392
Total megabyte-milliseconds taken by all reduce tasks=22435840
```

#### Map-Reduce Framework

```
Map input records=1546
Map output records=1546
Map output bytes=43288
Map output materialized bytes=46560
Input split bytes=1350
Combine input records=0
Combine output records=0
Reduce input groups=99
Reduce shuffle bytes=46560
Reduce input records=1546
Reduce output records=99
Spilled Records=3092
Shuffled Maps =30
Failed Shuffles=0
Merged Map outputs=30
GC time elapsed (ms)=4007
CPU time spent (ms)=12700
Physical memory (bytes) snapshot=6725951488
Virtual memory (bytes) snapshot=45217533952
Total committed heap usage (bytes)=5027065856
```

#### Shuffle Errors

```
BAD_ID=0
CONNECTION=0
IO_ERROR=0
WRONG_LENGTH=0
WRONG_MAP=0
WRONG_REDUCE=0
```

#### File Input Format Counters

```
Bytes Read=231793
```

#### File Output Format Counters

```
Bytes Written=2825
```

```
20/11/03 18:18:48 INFO streaming.StreamJob: Output directory: /output
```

```
A520M-4750G@cluster-d98c-m:~/hw3$
```

### 1.5. 執行 "cat result.dat" 查看結果

```
A520M-4750G@cluster-d98c-m:~/hw3$ cat result.dat
2004-03-07 T 16:00:00.000      27
2004-03-07 T 17:00:00.000      25
2004-03-07 T 18:00:00.000      24
2004-03-07 T 19:00:00.000      26
2004-03-07 T 20:00:00.000      20
2004-03-07 T 21:00:00.000      23
2004-03-07 T 22:00:00.000      29
2004-03-07 T 23:00:00.000      22
2004-03-08 T 00:00:00.000      21
2004-03-08 T 01:00:00.000      21
2004-03-08 T 02:00:00.000      27
2004-03-08 T 03:00:00.000      22
2004-03-08 T 04:00:00.000      26
2004-03-08 T 05:00:00.000      37
2004-03-08 T 06:00:00.000      17
2004-03-08 T 07:00:00.000      31
2004-03-08 T 08:00:00.000      44
2004-03-08 T 09:00:00.000      63
2004-03-08 T 10:00:00.000      39
2004-03-08 T 11:00:00.000      34
2004-03-08 T 12:00:00.000      45
2004-03-08 T 13:00:00.000      37
2004-03-08 T 14:00:00.000      23
2004-03-08 T 15:00:00.000       9
2004-03-08 T 16:00:00.000       2
2004-03-08 T 17:00:00.000       2
2004-03-08 T 18:00:00.000       9
2004-03-08 T 19:00:00.000       6
2004-03-08 T 20:00:00.000      23
2004-03-08 T 22:00:00.000      20
2004-03-08 T 23:00:00.000       1
2004-03-09 T 01:00:00.000      12
```

### 1.6. reference:

- 1.6.1. <https://jerryneest.io/dataproc-hadoop/>
- 1.6.2. <https://jerryneest.io/install-hadoop-on-ubuntu/>

1.6.3. <https://jerryneal.io/hadoop-map-reduce-apache2-web-server-access-log/>

2. (b) (10%) Write the Java version of part (a)

2.1. hourCount.java:

```
17 public class hourCount {
18
19     public static class TokenizerMapper
20         extends Mapper<Object, Text, Text, IntWritable>{
21
22         private final static IntWritable one = new IntWritable(1);
23         private Text word = new Text();
24
25         public void map(Object key, Text value, Context context
26             ) throws IOException, InterruptedException {
27             /*StringTokenizer itr = new StringTokenizer(value.toString());
28             while (itr.hasMoreTokens()) {
29                 word.set(itr.nextToken());
30                 context.write(word, one);
31             }*/
32
33             String[] temp = value.toString().split("\\[");
34             if(temp.length > 1){
35                 temp = temp[1].split("\\s+");
36                 SimpleDateFormat sdf_input = new SimpleDateFormat("dd/MMM/yyyy:HH:mm:ss");
37                 SimpleDateFormat sdf_output = new SimpleDateFormat("yyyy-MM-dd HH:00:00.000");
38
39                 try {
40                     word.set(sdf_output.format(sdf_input.parse(temp[0])));
41                     context.write(word, one);
42                 } catch (ParseException e) {
43                     ;
44                 }
45             }
46         }
47     }
48 }
```

```

49 public static class IntSumReducer
50     extends Reducer<Text,IntWritable,Text,IntWritable> {
51     private IntWritable result = new IntWritable();
52
53     public void reduce(Text key, Iterable<IntWritable> values,
54                       Context context
55                       ) throws IOException, InterruptedException {
56         int sum = 0;
57         for (IntWritable val : values) {
58             sum += val.get();
59         }
60         result.set(sum);
61         context.write(key, result);
62     }
63 }
64
65 public static void main(String[] args) throws Exception {
66     Configuration conf = new Configuration();
67     Job job = Job.getInstance(conf, "word count");
68     job.setJar("hc.jar");
69     job.setJarByClass(hourCount.class);
70     job.setMapperClass(TokenizerMapper.class);
71     job.setCombinerClass(IntSumReducer.class);
72     job.setReducerClass(IntSumReducer.class);
73     job.setOutputKeyClass(Text.class);
74     job.setOutputValueClass(IntWritable.class);
75     FileInputFormat.addInputPath(job, new Path(args[0]));
76     FileOutputFormat.setOutputPath(job, new Path(args[1]));
77     System.exit(job.waitForCompletion(true) ? 0 : 1);
78 }
79 }

```

2.2. 送上 cluster 後，依序執行：

- 2.2.1. hdfs dfs -mkdir /input
- 2.2.2. hdfs dfs -put access.log /input
- 2.2.3. hdfs dfs -rm -r -f /output
- 2.2.4. javac hourCount.java -cp \$(hadoop classpath)
- 2.2.5. jar cf hc.jar hourCount\*.class
- 2.2.6. hadoop jar hc.jar hourCount /input /output
- 2.2.7. hdfs dfs -cat /output/\* | sort -k1,1



```
A520M-4750G@cluster-d98c-m:~/hw3/java_hour_count$ hdfs dfs -cat /output/* | sort -k1,1
2004-03-07 16:00:00.000 27
2004-03-07 17:00:00.000 25
2004-03-07 18:00:00.000 24
2004-03-07 19:00:00.000 26
2004-03-07 20:00:00.000 20
2004-03-07 21:00:00.000 23
2004-03-07 22:00:00.000 29
2004-03-07 23:00:00.000 22
2004-03-08 00:00:00.000 21
2004-03-08 01:00:00.000 21
2004-03-08 02:00:00.000 27
2004-03-08 03:00:00.000 22
2004-03-08 04:00:00.000 26
2004-03-08 05:00:00.000 37
2004-03-08 06:00:00.000 17
2004-03-08 07:00:00.000 31
2004-03-08 08:00:00.000 44
2004-03-08 09:00:00.000 63
2004-03-08 10:00:00.000 39
2004-03-08 11:00:00.000 34
2004-03-08 12:00:00.000 45
2004-03-08 13:00:00.000 37
2004-03-08 14:00:00.000 23
2004-03-08 15:00:00.000 9
2004-03-08 16:00:00.000 2
2004-03-08 17:00:00.000 2
2004-03-08 18:00:00.000 9
2004-03-08 19:00:00.000 6
2004-03-08 20:00:00.000 23
2004-03-08 22:00:00.000 20
2004-03-08 23:00:00.000 1
2004-03-09 01:00:00.000 12
2004-03-09 02:00:00.000 15
2004-03-09 03:00:00.000 1
```

2.3. reference: <https://jerryneal.io/hadoop-wordcount-example/>

3. github url:  
<https://github.com/a2134666/CloudComputingHW3>